2.6 New Priorities in Assigning Porosities

In standard TOUGH2, porosity is specified through variable POR in block ROCKS. This value is assigned to all gridblocks which belong to the corresponding rock type. However, this porosity value can be overwritten on a gridblock-to-gridblock basis through variable PORX specified in block INCON. If porosity is one of the parameters to be estimated by inverse modeling, the porosity should be adjusted during the optimization, i.e., the porosity estimate provided by ITOUGH2 must have the highest priority, overwriting values stored in POR and PORX.

In order to test the implementation of this concept, we used three different ways to assign porosity to gridblocks "ELM 1", "ELM 2", and "ELM 3" as shown in file *vv* (Figure 2.3.1) and *vvi* (Figure 2.5.1). The initial guess for porosity specified in the ITOUGH2 input file is different from the corresponding one in the TOUGH2 input file, affecting "ELM 2". Porosity values are also given in block INCON for gridblocks "ELM 1" and "ELM 2". The porosities given in SAVE file *vv.sav* (Figure 2.6.1) reflect the values actually used in the simulation.

A summary is given in Table 2.6.1. The porosity value from block INCON has overwritten that from block ROCKS, and the porosity given in the ITOUGH2 input file has overwritten that from block INCON, in agreement with the intended behavior and thus fulfilling Requirement 6.

```
INCON -- INITIAL CONDITIONS FOR    3 ELEMENTS AT TIME  0.100000E+01
ELM 1           0.40000000E+00
 0.1077494458364E+06 0.1030002349086E+02 0.4999944736214E+02
ELM 2           0.60000000E+00
 0.1013411139066E+06 0.1029998794249E+02 0.2000110000267E+02
ELM 3           0.30000000E+00
 0.1000491003565E+06 0.1029999300758E+02 0.2000000330247E+02
+++
    1    3    5 0.00000000E+00 0.10000000E+01
```

Figure 2.6.1. File *vv.sav* showing porosity values used during the simulation.

Table 2.6.1. Porosities Assigned and Actually Used

Gridblock	ROCKS	INCON	ITOUGH2	SAVE
ELM 1	0.1	0.4	-	0.4
ELM 2	0.2	0.5	0.6	0.6
ELM 3	0.3	-	-	0.3

2.7 Adjusting Array Dimensions

Problems solved by ITOUGH2 vary considerably in size, depending on the number of gridblocks and connections used for discretization, the number of equations solved, the number of parameters estimated, the number of observations available, etc. Due to the overall size of ITOUGH2, it is important to be able to adjust the dimensions of major TOUGH2 and ITOUGH2 arrays to make the code fit on a specific computer with limited memory. Because ITOUGH2 is written in FORTRAN77, no dynamic memory allocation is possible, i.e., arrays are redimensioned by changing their size in the source code, followed by recompilation.

The design and architecture of ITOUGH2 allows for safe, convenient, and fast adjustment of major arrays. The purpose of this section is to prove that changing array dimensions using the procedure described herein does not corrupt the code.

The ITOUGH2 design makes use of the following features to assure safe maintenance of the code:

(1) All COMMON blocks holding major arrays are stored in INCLUDE files, making sure that any modification (such as redimensioning) is made consistently throughout the code.

(2) Array dimensions are given by constants, which are defined using PARAMETER statements. This assures consistency of arrays that must have identical dimensions. The PARAMETER statements are summarized in an INCLUDE file named *maxsize.inc*.

(3) Compilation is performed by means of a makefile and the "make" utility, available on UNIX machines and most PCs. This assures that all files affected by a change are recompiled.

(4) Checks are made within ITOUGH2 to assure that a given array is sufficiently large to accommodate the problem at hand. If an array index is greater than the size of the array, an error message is printed and ITOUGH2 run is stopped.

(5) The array dimensions used for a specific run are reported in output files for traceability (see Section 2.8).

The procedure for redimensioning major arrays can be described as follows:

(1) If an ITOUGH2 array is not sufficiently dimensioned, an error message is issued, indicating the constant that must be increased.

(2) The user edits file *maxsize.inc*, adjusting the appropriate constants.

(3) The user types "`make`" to recompile and relink ITOUGH2.

The following test runs assure that (A) ITOUGH2 cannot be run if an array is insufficiently dimensioned, and (B) if ITOUGH2 runs, its arrays are sufficiently dimensioned.

In order to perform Test A, the constants defined in file *maxsize.inc* were stepwise reduced until an error message was issued when running the recompiled code using the following command:

```
itough2 -v 3.2 vvRITi vvRIT 3 &
```

An example of an error message is shown in Figure 2.7.1.

The constants were then increased by 1 above the values that triggered the error message, yielding the minimum array sizes accepted by ITOUGH2. The corresponding file (named *minsize.inc*) is shown in Figure 2.7.2.

ITOUGH2 was then recompiled using minimum array dimensions, and a compiler option, which detects array size violations during compilation and execution. The on-line manual pages for the corresponding compiler option for the SUN Solaris 2 compiler f77, Version FORTRAN 77, SC4.2, are reproduced in Figure 2.7.3.

Adjusting array dimensions can be considered safe if ITOUGH2 compiles and runs properly with array dimensioned minimally for the given test problem, since array range violations are most rigorously detected with minimum array dimensions. If arrays are larger than the problem size, no problems are expected to occur. If array dimensions are too large to make ITOUGH2 fit in the computer's memory, either the code cannot be run, or its speed performance deteriorates. Neither case poses a risk that erroneous simulation results are obtained.

ITOUGH2 could be compiled and run with minimum array dimensions, fulfilling Requirement 7.

```
*****  ERROR  *****
* Number of parameters exceeds MAXN =  2.
* Increase MAXN in file maxsize.inc and recompile!
*****  ERROR  *****
```

Figure 2.7.1. Excerpt from ITOUGH2 output file *vvRITi.out*, show error message if arrays are insufficiently dimensioned.

```
C$$$$$$$$$ PARAMETERS FOR SPECIFYING THE MAXIMUM PROBLEM SIZE $$$$$$$$$$
C
C  *********************************************************************
C ITOUGH2 and TOUGH2 parameter statements
C  *********************************************************************
C
C --- MAXTIM   : Maximum number of calibration times
       INTEGER MAXTIM
       PARAMETER (MAXTIM=61)
C
C  *********************************************************************
C TOUGH2 parameter statements
C  *********************************************************************
C
C --- MAXEL    : Maximum number of elements
       INTEGER MAXEL
       PARAMETER (MAXEL=53)
C
C --- MAXCON   : Maximum number of connections
       INTEGER MAXCON
       PARAMETER (MAXCON=52)
C
C --- MAXK     : Maximum number of components/species
       INTEGER MAXK
       PARAMETER (MAXK=2)
C
C --- MAXEQ    : Maximum number of equations per block
       INTEGER MAXEQ
       PARAMETER (MAXEQ=3)
C
C --- MAXPH    : Maximum number of phases
       INTEGER MAXPH
       PARAMETER (MAXPH=2)
C
C --- MAXB     : Maximum number of phase-dependent secondary variables
C               other than component mass fractions
       INTEGER MAXB
       PARAMETER (MAXB=6)
C
C --- MAXSS    : Maximum number of sources/sinks
       INTEGER MAXSS
       PARAMETER (MAXSS=1)
C
C --- MAVTAB   : Maximum average number of table entries per sink/source
       INTEGER MAVTAB
       PARAMETER (MAVTAB=1)
C
C --- MAXROC   : Maximum number of rock types
       INTEGER MAXROC
       PARAMETER (MAXROC=3)
C
C --- MAXTSP   : Maximum number of specified time steps divided by 8
       INTEGER MAXTSP
       PARAMETER (MAXTSP=1)
C
C --- MAXLAY   : Maximum number of reservoir layers for deliverability
       INTEGER MAXLAY
       PARAMETER (MAXLAY=1)
C
C --- MAXRPCP : Maximum number of parameters for a relative permeability
C                or a capillary pressure function
C                (to get more than 7, more input lines may be needed!).
       INTEGER MXRPCP
       PARAMETER (MXRPCP=7)
C
C --- MXPCTB   : Maximum points in table of ECM capillary pressure
C                vs. saturation
       INTEGER MXPCTB
       PARAMETER (MXPCTB=1)
C
C --- MXTBC    : Maximum number of elements with time vs. boundary condition
C --- MXTBPT   : Maximum number of time vs. pressure data
       INTEGER MXTBC,MXTBPT
       PARAMETER (MXTBC=1)
       PARAMETER (MXTBPT=1)
```

Figure 2.7.2. File *maxsize.inc* with minimum array dimensions for test problem.

```
C --- Storage for MA28.  LIRN is the size of IRN and needs to be larger
C      than the number of non-zeros NZ=(NEL+2*NCON)*NEQ*NEQ.
C      LICN is the length of ICN and CO.
       INTEGER LICN,LIRN
        PARAMETER (LIRN=2*(MAXEL+2*MAXCON)*MAXEQ*MAXEQ)
        PARAMETER (LICN=4*(MAXEL+2*MAXCON)*MAXEQ*MAXEQ)
C
C --- Parameters for conjugate gradient package t2cg1
        INTEGER NREDM,MNZ,NRWORK,NIWORK
        PARAMETER (NREDM=MAXEQ*MAXEL)
        PARAMETER (MNZ=(MAXEL+2*MAXCON)*MAXEQ*MAXEQ)
        PARAMETER (NRWORK=1000+MNZ+38*NREDM)
        PARAMETER (NIWORK=32+MNZ+5*NREDM)
C
C --- Parameters for IFS
C      MAXIFSP  : Maximum number of IFS parameters
       INTEGER MAXIFSP
       PARAMETER (MAXIFSP=1)
C
C ***********************************************************************
C ITOUGH2 parameter statements
C ***********************************************************************
C --- MAXN      : Maximum number of parameters to be estimated
       INTEGER MAXN
       PARAMETER (MAXN=3)
C
C --- MAXO      : Maximum number of datasets
        INTEGER MAXO,MAXOTWO
       PARAMETER (MAXO=2)
        PARAMETER (MAXOTWO=2*MAXO)
C
C --- MAXM      : Maximum number of calibration points
C      (approx. number of datasets times number of calibration times)
       INTEGER MAXM
       PARAMETER (MAXM=123)
C
C --- MAXPD     : Max number of paired data
       INTEGER MAXPD
       PARAMETER (MAXPD=120)
C
C --- MAXR      : Dimension of array RPAR and IPAR, ROBS and IOBS
       INTEGER MAXR
       PARAMETER (MAXR=10)
C
C --- MAXBRK    : Max number of points in time at which SAVE file is written (restart)
       INTEGER MAXBRK
       PARAMETER (MAXBRK=1)
C
C --- MAXEBRK   : Max number of elements with new initial conditions after break
       INTEGER MAXEBRK
       PARAMETER (MAXEBRK=1)
C
C --- MAXCOEFF : Max number of coefficients for data modeling functions
       INTEGER MAXCOEFF
       PARAMETER (MAXCOEFF=1)
C
C --- MAXMCS    : Max number of Monte Carlo simulations
       INTEGER MAXMCS
       PARAMETER (MAXMCS=1)
C
C --- MAXCURVE : Max number of curves to be plotted
       INTEGER MAXCURVE
       PARAMETER (MAXCURVE=6)
```

Figure 2.7.2. (cont.) File *maxsize.inc* with minimum array dimensions for test problem.

```
F77(1)                  User Commands                          F77(1)

NAME
     f77 - FORTRAN 77 compiler

...

DESCRIPTION
     f77 is a superset of FORTRAN 77.
     Version:  FORTRAN 77 SC4.2

...

     -C    Check array references for out of range subscripts.

           Subscripting arrays beyond their declared sizes may
           result in unexpected results, including segmentation
           faults. The -C option checks for possible array sub-
           script violations in the source code and during execu-
           tion.

           If the -C option is used, array subscript violations
           are treated as an error. If an array subscript range
           violation is detected in the source code during compi-
           lation, it is treated as a compilation error.

           This option will increase the size of the executable
           file.
```

Figure 2.7.3. Manual pages for compiler option `-C`, checking array subscript violations.

2.8 Application Control

The application control of ITOUGH2 simulations was enhanced to improve traceability. The following information is printed to either the TOUGH2 output file, the ITOUGH2 output file, or the ITOUGH2 message file:

- Starting and ending date and time of run;
- Names of TOUGH2 and ITOUGH2 input files;
- Directory name of input and output files;
- Equation-of-state module used;
- Name of script file used to run ITOUGH2;
- Command arguments passed to script file;
- Name of ITOUGH2 executable;
- Type of computer used;
- Computer host name;
- Login name of user;
- Constants used for dimensioning of major arrays (see Section 2.7);
- Version control statements for each subroutine.

Figures 2.8.1 through 2.8.4 show various excerpts of the ITOUGH2 output file *vvRITi.out*. Note the correct reporting of command line arguments, and the array dimension statements, which agree with the values given in include file *maxsize.inc*, shown in Figure 2.7.2. Requirement 8 is considered fulfilled.